Tip of the week

Logging the linear digital potentiometer

Small, resilient and easy to use, digital potentiometers are becoming widely accepted as replacements for mechanical potentiometers, especially in volume-control applications. In this type of audio application, it is often preferable to use a log-taper pot instead of a linear-taper pot because the human ear perceives the logarithmic attenuation of sound as linear attenuation. But if all that is available for volume control is a high-resolution linear potentiometer, all is not lost. The following analysis shows how to use a linear digital pot as a log pot.

Because a potentiometer is essentially a voltage divider, its output voltage V_{OUT} can be written in terms of the voltage applied to its input V_{IN} (if V_{IN} is applied at R_H):

Equation 1: $V_{OUT} = V_{IN} x(R_{W-L}/R_{H-L})$

By Joe Neubauer (Joe_Neubauer@maximhq.com), *a technical business manager,* and Alex Shih (Alex_Shih@maximhq.com), *a member of the technical staff, at Maxim Integrated Products Inc. (Sunnyvale, Calif.).*

where R_{W-L} is the resistance between the wiper (W) and the low end of the resistor string (L) and R_{H-L} is the total end-to-end resistance of the resistor string.

The attenuation of a given signal in decibels can be found using the following equation:

Equation 2: Attenuation (dB) = 20log (V_{OUT}/V_{IN})

Substituting the V_{OUT} value from equation (1) into equation (2) yields:

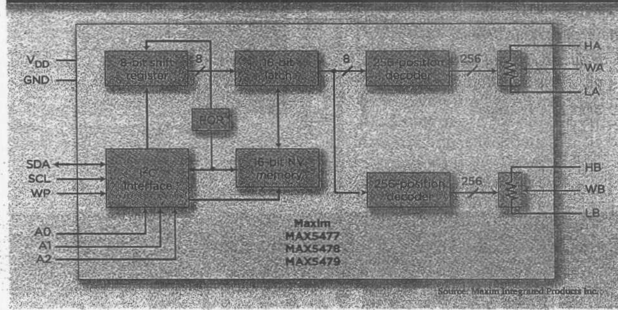
Equation 3: Attenuation (dB) = 20log$[V_{IN}(R_{W-L}/R_{H-L})/V_{IN}]$ = 20log(R_{W-L}/R_{H-L})

But in a linear potentiometer, tap points are located at equal resistor segments along an entire resistor string and these resistances can be represented by the tap position (R_{W-L}) and the total number of taps (R_{H-L}). The tap position (R_{W-L}) can, therefore, be represented as:

Equation 4: (R_{W-L}) = (Total_Taps – x)R

Digital pots line takes standard configuration

It includes a link to both high and low ends of resistor string



Source: Maxim Integrated Products Inc.

where x = 1, 2, 3. . . Total_Taps.

The total end-to-end resistance can be represented as R_{H-L} = (Total_Taps – 1)xR because the first tap point (minimum attenuation) contains no resistance from the resistor string. Now the attenuation equation takes the form of:

Equation 5: Attenuation (dB) = 20log[(Total_Taps – x)/(Total_Taps – 1)]

where x ≠ Total_Taps. If x = Total_Taps, then the attenuation approaches $-\infty$.

Thus the linear-taper digital potentiometer can convert to a logarithmic-taper potentiometer. This works best with higher-resolution linear pots (128 taps or greater) because logarithmic resolution is severely limited for linear pots with fewer taps. Also, other internal structures my cause some inaccuracies at the endpoints. The accompanying source code converts desired attenuation to tap position as well as desired tap position to the corresponding attenuation.

Converting attenuation to tap position

And Vice versa

```
//------------------------------------------------------------------------

// __int16 DB_to_Tap(double dB, __int16 TotalTaps);
// input: dB -- double, db should be less than or equal to 0
//          TotalTaps -- __int16, TotalTaps should bigger than 0
// output: Function will returns the Tap number as __int16

__int16 DB_to_Tap(double dB, __int16 TotalTaps)
{
    __int16 Closest_Tap;
    double Raw_Value;


    if(dB > 0)   // Check the input
    {
        Closest_Tap = 1;
    }
    else
    {
       // Convert to the closest Tap number
        Raw_Value = TotalTaps - (pow(10, (dB/20)) * (TotalTaps - 1));
        Closest_Tap = (__int16)(Raw_Value + 0.5);
    }

    return(Closest_Tap);
}
//------------------------------------------------------------------------

//------------------------------------------------------------------------

// double Tap_to_DB(__int16 Tap, __int16 TotalTaps);
// input: Tap -- __int16, Tap should be greater then 0 and less than TotalTaps.
//                   If Tap equal to TotalTaps, the result should be negative infinity
//          TotalTaps -- __int16, TotalTaps should bigger than 0
// output: Function will returns the dB value as double

double Tap_to_DB(__int16 Tap, __int16 TotalTaps)
{
    __int16    tmp_a;
    double dB_Value, tmp_b;

    // Check input value
    tmp_a = Tap;
    if(Tap >= TotalTaps) tmp_a = TotalTaps - 1;
    if(Tap < 1)               tmp_a = 1;

    // Convert to dB value
    tmp_b = (double)(TotalTaps - tmp_a);
    tmp_b = tmp_b / (TotalTaps - 1);
    dB_Value = 20 * log10(tmp_b);

    return(dB_Value);
}
//------------------------------------------------------------------------
```

Source: Maxim Integrated Products Inc.